Proceedings of SAT Competition 2025: Solver and Benchmark Descriptions, Proceedings of SAT Competitions, Vienna, 2025.

Split-BDD Multiplier Verification Benchmarks Submitted to the SAT Competition 2025

Armin Biere*, Alexander Konrad*, Daniela Kaufmann†, Christoph Scholl*
* *University of Freiburg*, Germany † *TU Wien*, Austria

Formal verification of arithmetic circuits is an important problem but industrial practice still requires a substantial amount of manual effort. Recently an approach based on automated reasoning with polynomials using computer-algebra promises to scale much better than previous automated approaches, but still has issues with heavily optimized circuits.

In this context the verification of integer multiplier circuits serves as litmus test for the viability of such an approach. This is also our target here. For an overview on recent results and further pointers on integer multiplier verification see [1]–[4].

One idea is to use plain old reduced order *bit-level* binary decision diagrams (BDDs) for verifying integer multipliers, which, at first sight, looks impossible as building BDDs for integer multipliers is well known to explode [5]. However already Burch showed in 1991 [6], that splitting variables of a BDD, i.e., using new copies for each output slice, yields polynomial "split" BDDs. This idea has recently been revisited in [7]. The authors also give an algorithm for efficiently constructing a split-BDD specification for a given bit-width, which then can serve as *golden model* for an implementation.

The down-side of this approach we follow, is, first, that building the BDD for a given implementation, referred to as *symbolic simulation*, might still explode and, second, that the resulting split-BDD is not canonical and needs to be compared to the golden split-BDD. For this last task we build the XORs of BDDs of individual simulated and golden split-BDDs and then use a SAT solver to show that all of theses are *false*.

Thus our benchmarks encode the problem of showing the equivalence of the result of symbolic simulation of a gate-level optimized integer multiplier circuit to the golden split-BDD.

Using the GENMUL multiplier generator [8] we generated 252 Verilog models for 28 multiplier architectures for 9 bit-widths, from 16 to 48 with an increment of 4. Those were then synthesized with Yosys [9] into AIGs and further optimized with ABC [10] by applying the synthesis script "`resyn; resyn2; resyn3`" until fix-point.

Our internal prototype verification tool AIGAINDDS implements a variant of the approach presented in [7] and follows the procedure outlined above. At the end it yields a CNF for the resulting miter. In the CNF encoding we use 6 clauses for each node of the BDD to increase arc-consistency.

For specific multiplier architectures and larger bit-widths our symbolic simulation explodes despite splitting variables. We therefore had to limit main memory to 60 GB and in total produced 128 CNFs. However, it turned out that partial normalization inherent in the split-BDD approach produced many identical intermediate split-BDDs as well as CNFs and only 27 unique CNFs remain: 7 for bit-width 16, 6 for 20, 4 for 24, 3 for 28 and 32, and 1 for each of 36, 40, 44, 48. The benchmarks are named as "gm*n* sp*at fa*.cnf", with *n* the bit-width, *at* the architecture of the adder tree and *fa* of the final stage adder, e.g., "gm48sparrc.cnf" is a CNF for an architecture (array adder tree + ripple-carry final stage adder) for which we were able to produce CNFs for all bit-widths.

We evaluated these benchmarks with our award winning SAT solver KISSAT [11] and MINISAT. From the 27 benchmarks submitted both solved the same 10 benchmarks within one hour, with MINISAT being slightly faster.

This work was supported in part by the state of Baden-Württemberg through bwHPC, the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG.

REFERENCES

[1] A. Mahzoon, D. Große, and R. Drechsler, "Revsca-2.0: Sca-based formal verification of nontrivial multipliers using reverse engineering and local vanishing removal," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 41, no. 5, pp. 1573–1586, 2022.

[2] D. Kaufmann and A. Biere, "Improving AMulet2 for verifying multiplier circuits using SAT solving and computer algebra," *Int. J. Softw. Tools Technol. Transf.*, vol. 25, no. 2, pp. 133–144, 2023.

[3] A. Konrad and C. Scholl, "Symbolic computer algebra for multipliers revisited - it's all about orders and phases," in *Formal Methods in Computer-Aided Design (FMCAD'24)*. IEEE, 2024, pp. 261–271.

[4] R. Li, L. Li, H. Yu, M. Fujita, W. Jiang, and Y. Ha, "RefSCAT: formal verification of logic-optimized multipliers via automated reference multiplier generation and SCA-SAT synergy," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 44, no. 2, pp. 791–804, 2025.

[5] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. Computers*, vol. 35, no. 8, pp. 677–691, 1986.

[6] J. R. Burch, "Using BDDs to verify multipliers," in *Proc. 28th Design Automation Conference (DAC'91)*. ACM, 1991, pp. 408–412.

[7] J. Kumar, Y. Miyasaka, A. Srivastava, and M. Fujita, "Formal verification of integer multiplier circuits using binary decision diagrams," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 42, no. 4, pp. 1365–1378, 2023.

[8] A. Mahzoon, D. Große, and R. Drechsler, *GenMul: Generating Architecturally Complex Multipliers to Challenge Formal Verification Tools*. Springer International Publishing, 2021, pp. 177–191.

[9] D. Shah, E. Hung, C. Wolf, S. Bazanski, D. Gisselquist, and M. Milanovic, "Yosys+nextpnr: An open source framework from Verilog to Bitstream for commercial FPGAs," in *27th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, (FCCM'19)*. IEEE, 2019, pp. 1–4.

[10] R. K. Brayton and A. Mishchenko, "ABC: an academic industrial-strength verification tool," in *Computer Aided Verification (CAV'10)*, ser. LNCS, vol. 6174. Springer, 2010, pp. 24–40.

[11] A. Biere, T. Faller, K. Fazekas, M. Fleury, N. Froleyks, and F. Pollitt, "CaDiCaL, Gimsatul, IsaSAT and Kissat entering the SAT Competition 2024," in *Proc. of SAT Competition 2024 – Solver, Benchmark and Proof Checker Descriptions*, ser. Department of Computer Science Report Series B, vol. B-2024-1. University of Helsinki, 2024, pp. 8–10.